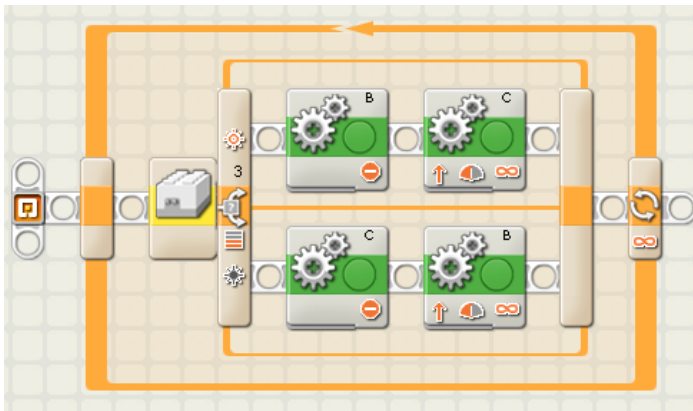


Title: Programming Guide for Molly the Meerkat

I really saw this project broken down into three parts but I don't necessarily think they are three distinct parts. First, the idea of what I wanted to happen had to occur. My environment had to propel this thinking. So the environment was a big part of how I built my robot. For instance, the maximum height of the head in its raised position and the minimum height of the head in lowered position drove how tall my tunnel's ceilings could be. Third, the programming had to be worked on in steps as the environment was built because I wanted to see if the robot would be able to act accordingly. There were many challenges in programming that had to be tackled and I tried to solve them individually. The individual challenges in programming were; getting the line follower programmed to work on reflected light, having another light sensor search for light from the "sky", getting the head to raise and lower twice successfully without tearing itself apart, and using a sound sensor to react in a specific way to a sound. After I can get the robot to work in small stages, I would try to put the program all together.

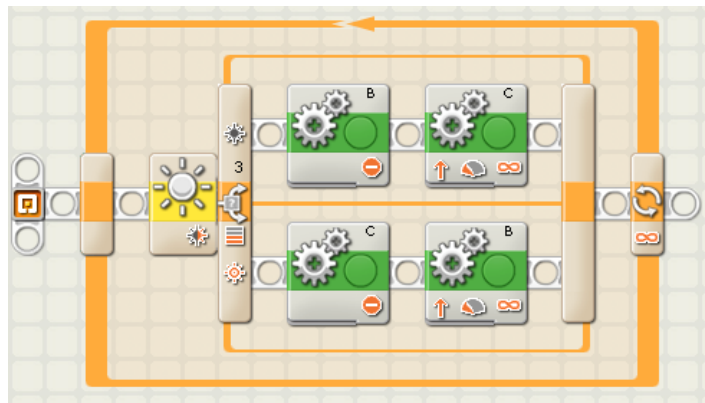
I. Getting the line follower programmed to work on reflected light. I started out with the old RCX light sensor to act as my line follower sensor. When I placed the RCX light sensor on the white duct tape that the robot had to follow I got a reading of 59 and placement on the brownish back of the 1/8 inch hardboard tunnel "floor" yielded a reading of 45. So I set the threshold at 52. But as I started my programming I noticed that the RCX light sensor did not have the option for generate light as the NXT sensor did. So I had to see if the RCX light sensor would follow the line in pitch black by generating its own light.



When this program ran the robot simply turned circles and I noticed that no light was being emitted by the RCX light sensor. So back to the drawing board. I'll have to put the NXT sensor on for the line follower and hope the RCX can see the light from the hole in the ceiling.

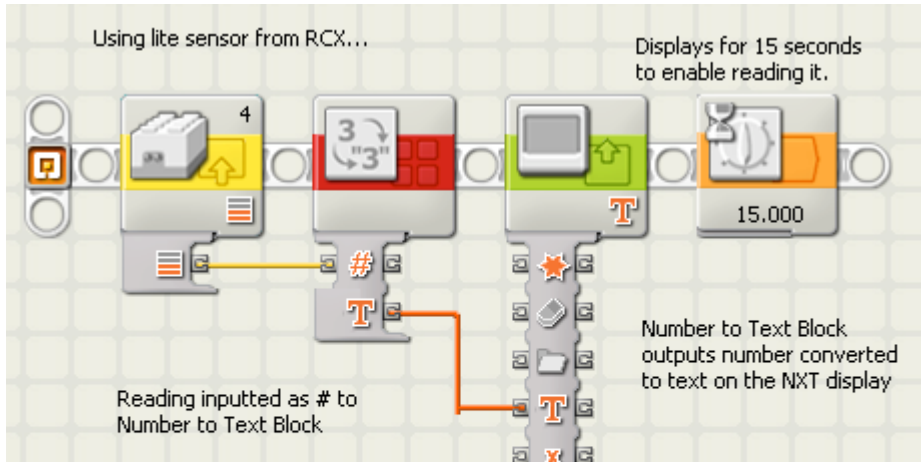
I had to change the program to the NXT light sensor and I set it for a threshold of halfway between 60 and 42. (51)

This program worked with a threshold at 51 and the power of both B and C engine reduced to 25%. A little slow for a scampering Molly the meerkat through her burrows in Africa, but she's just eaten and she's a little slow.



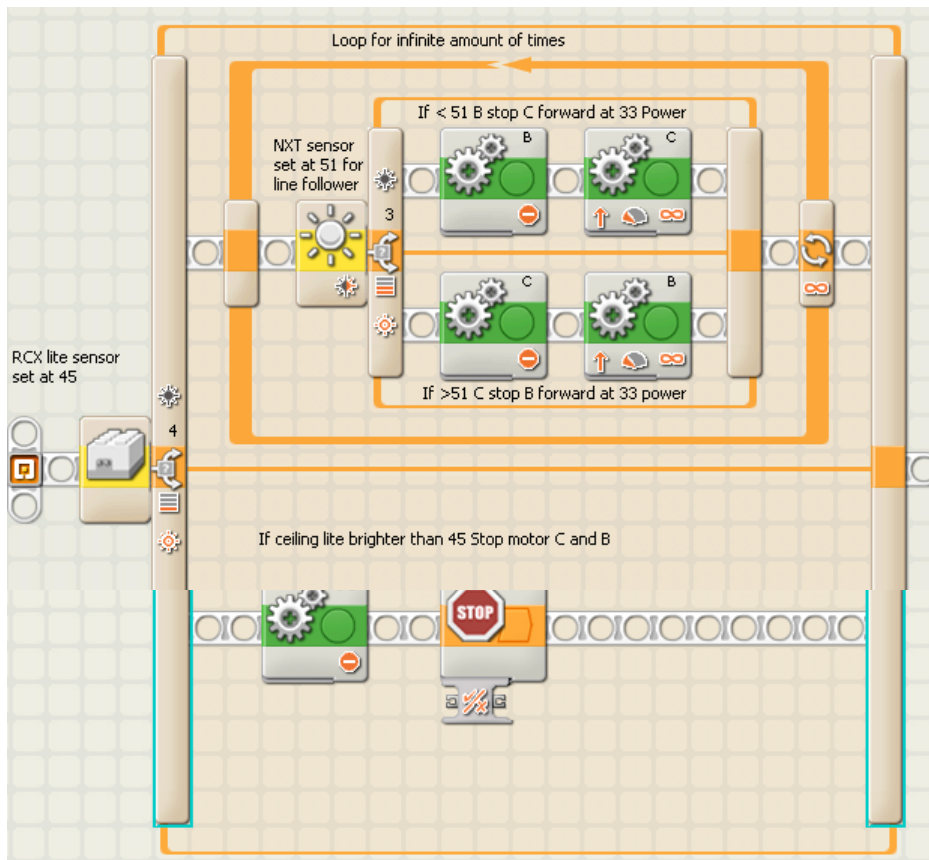
II. Having another light sensor search for sky light.

I had to write a quick program (JG Ceiling Check) to enable me to put the meerkat robot into the darkened tunnel, and read both the darkness level of the tunnel ceiling and the lightness level of the hole that the meerkat will encounter. I used the mining robot programming on volume 2 as my guide.



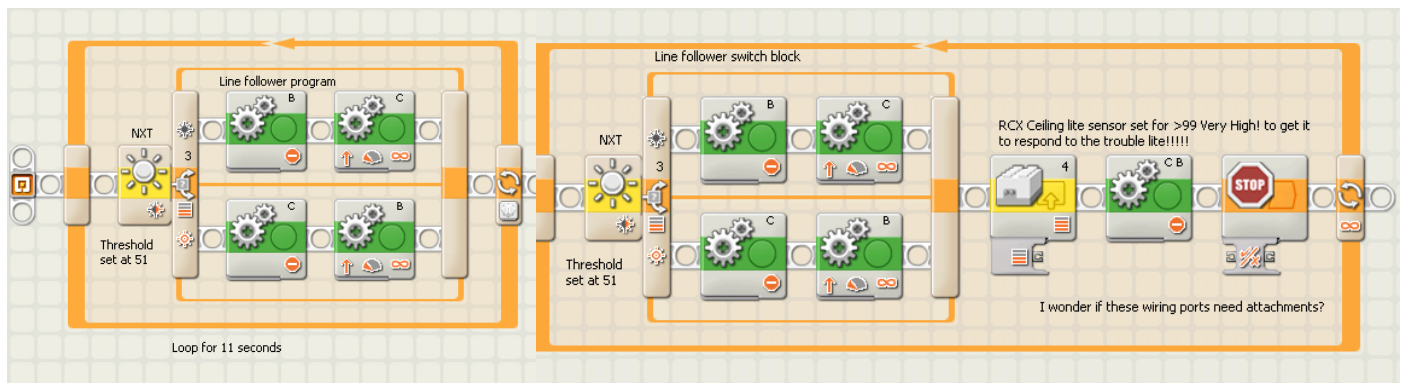
The dark level read inconsistently but most times 31 and the light level read quite variably; with normal garage light (fluorescent 12 feet high) 59; with a trouble light (about a foot over the tunnels) 100. I knew I was going to have some problems trying to get a consistent read.

I then tried to add into the line follower program another that would stop the motors upon seeing the light from the hole in the ceiling.



The result of the program (Meerkat 1) was that the robot followed the line but never stopped, even when I held the trouble lite less than an inch over the hole. The RCX lite sensor was emitting a lite.

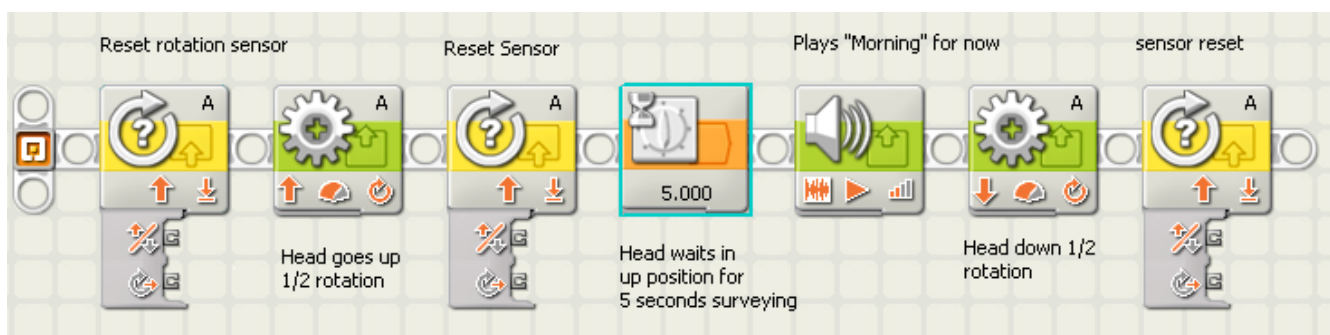
So, on to (Meerkat 2) to attempt a 2nd try. I thought perhaps I'm not using the loops quite right. I tried a more linear process.



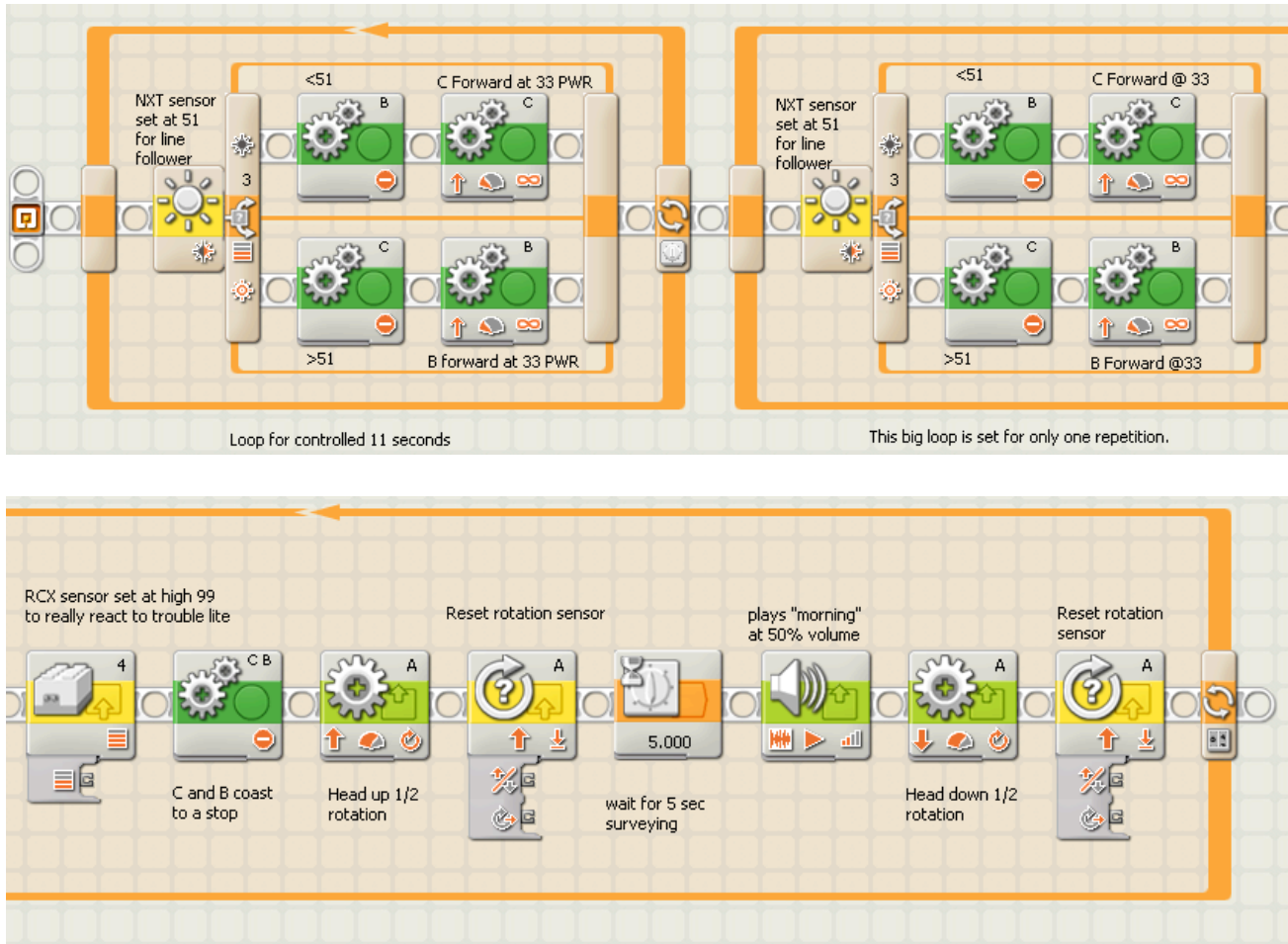
The above was called (Meerkat 2) and it didn't work either. I added the first loop of line follower to the front for a total time of 11 seconds so that the robot wouldn't stop immediately upon being released into the tunnels. This hopefully would get it going then start trying to detect the ceiling lite. This worked up to the first hole in the ceiling, so I moved on to the Head Raise and Lower Program.

III. Getting the head to raise and lower twice successfully without tearing itself

apart. I used a liberal dose of resetting the rotation sensor so I wouldn't accidentally go too far with the motor and tear something up. The worked well, and the stability of the head wavering sideways has not been an issue yet. Now I've got to incorporate the (Head Up) program into my (Meerkat 2) program and see what else fails.



Now combining (Head Up) program into my (Meerkat 2) turned into (Meerkat 3). Which is shown below.



The result was that (Meerkat 3) worked to get the robot to stop under the first hole, says “morning” (until I can find an authentic sound), then shuts off. My dilemma is that I just don’t want to depend on the timing to get it going first off in the tunnel and I don’t want to have to do another timing just to get it to the second hole. I want it to react more closely to the environment.

The fourth attempt (Meerkat 4) was simply Meerkat 3 doubled in size, a 12 second line follower loop to get the robot into the darkness of the tunnel (otherwise it would shut off immediately, and the head would go up since it sensed the light of the entrance), a true infinite line follower loop until the ceiling lite stopped it and then the head went up and down. Then it would repeat the process, but several things were wrong with this. First, it shouldn’t have to depend on timing to work. What if my battery level was low. The trouble light messed up the line follower as it was shining on the base of the tunnel in a more concentrated area. I decided to scrap most of what I was doing and try to create a more efficient program!

The exasperating fifth attempt, (Meerkat 5) was built thought by thought by action by action. It also works, with only a couple more things to figure out. I also dedicate this to my wife, Shari, whose ability to listen to me talk through what I wanted the program to do enables her to say, “hey, what if you did this.” I also made some radical design changes in my programming, starting with the the first of the loop where the RXC lite sensor transmits the value of the light through the data hubs to a Range block (my first experience with one. Sorry, the next page’s program were taped together, that’s why the edges of blocks don’t fit well together.